IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR LETTERS PATENT

Systems and Methods for Limiting Access To Potentially Dangerous Code

Inventor(s):

Rico Mariani
David M. Broman
Sanjeev K. Rajan
Kristi L. Cooper

ATTORNEY'S DOCKET NO. MS1-579US

TECHNICAL FIELD

The systems and methods described herein relate to network security and, more particularly, to securing web pages and software controls to prevent unauthorized web pages from utilizing software controls on a client computer to corrupt or misappropriate data on the client computer.

BACKGROUND

Website developers frequently utilize software controls to provide specialized functionality to web applications. Generally, a software control (hereinafter, "control") is defined as program instructions that manage data-handling tasks. Controls are typically reusable software components in binary form that can be plugged into other software components with relatively little effort. For example, a stock ticker control may be used to add a live stock ticker to a web page, or an animation control can be used to add animation features to a web page.

Controls may be downloaded to a Client computer together with the web pages that invoke them. Once a control is downloaded by a web page, it remains on the Client computer. Subsequent execution of the web page will execute the control without requiring the control to be downloaded again. However, other web pages may also invoke the control, even though the control was not downloaded with that web page. This invocation may even be accomplished without the user's knowledge.

This can lead to exploitation of the control by an unauthorized user. The unauthorized user may use the control for something other than its intended function, or use the control function in a manner contrary to the intended use of

the control function. The results of such exploitation can be loss or corruption of data, exposure of sensitive materials, or other security compromises.

As an example of how serious this exploitation can be, consider a user who downloads a control that access banking software on the user's computer. The user trusts the author of the control and the website, and uses the control according to its intended function. But when the user has finished using the control, the user may not even be aware that the control and its functionality remain on the user's computer. Thereafter, a web page set up by a hacker and accessed by the user may invoke the control and gain access to the user's banking software. The hacker may then have the ability to write unauthorized checks on the user's account, transfer funds electronically from the account, and so on.

To help combat this problem, signed controls have been developed. Signed controls contain a digital signature that uniquely identifies the author of the control. When the signed control is accessed, the control is authenticated by the downloading computer. Once authenticated, a determination is made as to whether the author of the control is an authorized source for controls. If so, the control may be invoked. However, this verification is only made when the control is initially downloaded. Once the user downloads the control, the control may be invoked by any other application without authorization from the user.

In addition to signed controls, the notion of trusted sites has been utilized whereby a user may confidently use a control downloaded from certain user-identified sites. Again, however, the problem remains that once a user has authorized the download of a control, the user can no longer safeguard against unauthorized use of that control.

Some operating systems, such as the WINDOWS family of operating systems produced by MICROSOFT CORPORATION, provide a feature whereby a control writer can specifically mark a control as being "safe" to avoid having to perform additional steps each time the control is used. A control can only be marked as safe if no other web site could possibly use the control in an unsafe manner. Once the control is marked as safe, it can be invoked without further precautionary measures being taken.

It is desirable to mark a control as safe so that a computer user can be confident that the control can be downloaded without causing harm to the user's computer. However, many valuable controls that can be safely invoked cannot be marked as safe because they do not satisfy the requirement that they cannot be used in an unsafe manner. These controls must be marked as "unsafe" even though they can be invoked in a safe manner. This is problematic in that a user may not download such a control simply because it is marked as unsafe, since the user does not know the exact reason that the control has been marked as unsafe. Such an unsafe designation may cause unnecessary apprehension and inconvenience to the user.

The implementations described herein overcome this disadvantage and allow a control writer to mark a control as safe, since malicious web pages will be prevented from invoking the safe control in an unsafe manner.

SUMMARY

Methods and systems are described herein that allow a control to be invoked only by an authenticated and authorized application. A web page is described that invokes a software control that has been previously downloaded to a Client computer, or which is contained in the web page to be downloaded by the Client computer. The web page is digitally signed by the author so that the Client computer can ensure that the control is being invoked by a trusted source. A confirmation module located in a web browser on the Client computer or in the control itself authenticates the digital signature and confirms whether the web page is authorized by the Client computer to invoke the control. If the web page is authenticated and authorized, then the Client computer allows the web page to invoke the control.

The described implementations solve the problems presented above, because an invoking application is authenticated and authorized each time the control is invoked rather than only when the control is downloaded. Therefore, an unauthorized user cannot gain access to a control previously downloaded onto the Client computer.

BRIEF DESCRIPTION OF THE DRAWINGS

A more complete understanding of exemplary methods and arrangements of the present invention may be had by reference to the following detailed description when taken in conjunction with the accompanying drawings wherein:

Fig. 1 is a diagram of an exemplary computer system on which the described embodiments may be implemented.

Fig. 2 is a block diagram of a server computer and a client computer according to an implementation described herein.

Fig. 3 is a flow diagram of a process to prevent use of a control by an unauthorized application.

DETAILED DESCRIPTION

The invention is illustrated in the drawings as being implemented in a suitable computing environment. Although not required, the invention will be described in the general context of computer-executable instructions, such as program modules, to be executed by a computing device, such as a personal computer or a hand-held computer or electronic device. Generally, program modules include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types. Moreover, those skilled in the art will appreciate that the invention may be practiced with other computer system configurations, including multi-processor systems, microprocessor-based or programmable consumer electronics, network PCs, minicomputers, mainframe computers, and the like. The invention may also be practiced in distributed computing environments where tasks are performed by remote processing devices that are linked through a communications network. In

a distributed computing environment, program modules may be located in both local and remote memory storage devices.

Exemplary Computer Environment

The various components and functionality described herein are implemented with a number of individual computers. Fig. 1 shows components of typical example of such a computer, referred by to reference numeral 100. The components shown in Fig. 1 are only examples, and are not intended to suggest any limitation as to the scope of the functionality of the invention; the invention is not necessarily dependent on the features shown in Fig. 1.

Generally, various different general purpose or special purpose computing system configurations can be used. Examples of well known computing systems, environments, and/or configurations that may be suitable for use with the invention include, but are not limited to, personal computers, server computers, hand-held or laptop devices, multiprocessor systems, microprocessor-based systems, set top boxes, programmable consumer electronics, network PCs, minicomputers, mainframe computers, distributed computing environments that include any of the above systems or devices, and the like.

The functionality of the computers is embodied in many cases by computer-executable instructions, such as program modules, that are executed by the computers. Generally, program modules include routines, programs, objects, components, data structures, etc. that perform particular tasks or implement particular abstract data types. Tasks might also be performed by remote processing devices that are linked through a communications network. In a

distributed computing environment, program modules may be located in both local and remote computer storage media.

The instructions and/or program modules are stored at different times in the various computer-readable media that are either part of the computer or that can be read by the computer. Programs are typically distributed, for example, on floppy disks, CD-ROMs, DVD, or some form of communication media such as a modulated signal. From there, they are installed or loaded into the secondary memory of a computer. At execution, they are loaded at least partially into the computer's primary electronic memory. The invention described herein includes these and other various types of computer-readable media when such media contain instructions programs, and/or modules for implementing the steps described below in conjunction with a microprocessor or other data processors. The invention also includes the computer itself when programmed according to the methods and techniques described below.

For purposes of illustration, programs and other executable program components such as the operating system are illustrated herein as discrete blocks, although it is recognized that such programs and components reside at various times in different storage components of the computer, and are executed by the data processor(s) of the computer.

With reference to Fig. 1, the components of computer 100 may include, but are not limited to, a processing unit 120, a system memory 130, and a system bus 121 that couples various system components including the system memory to the processing unit 120. The system bus 121 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. By way of example, and not

limitation, such architectures include Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISAA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnect (PCI) bus also known as the Mezzanine bus.

Computer 100 typically includes a variety of computer-readable media. Computer-readable media can be any available media that can be accessed by computer 100 and includes both volatile and nonvolatile media, removable and non-removable media. By way of example, and not limitation, computer-readable media may comprise computer storage media and communication media. "Computer storage media" includes both volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information such as computer-readable instructions, data structures, program modules, or other data. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical disk storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can be accessed by computer 110. Communication media typically embodies computer-readable instructions, data structures, program modules or other data in a modulated data signal such as a carrier wave or other transport mechanism and includes any information delivery media. The term "modulated data signal" means a signal that has one or more if its characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, communication media includes wired media such as a wired network or direct-wired connection and wireless media such as acoustic, RF, infrared and

other wireless media. Combinations of any of the above should also be included within the scope of computer readable media.

The system memory 130 includes computer storage media in the form of volatile and/or nonvolatile memory such as read only memory (ROM) 131 and random access memory (RAM) 132. A basic input/output system 133 (BIOS), containing the basic routines that help to transfer information between elements within computer 100, such as during start-up, is typically stored in ROM 131. RAM 132 typically contains data and/or program modules that are immediately accessible to and/or presently being operated on by processing unit 120. By way of example, and not limitation, Fig. 1 illustrates operating system 134, application programs 135, other program modules 136, and program data 137.

The computer 100 may also include other removable/non-removable, volatile/nonvolatile computer storage media. By way of example only, Fig. 1 illustrates a hard disk drive 141 that reads from or writes to non-removable, nonvolatile magnetic media, a magnetic disk drive 151 that reads from or writes to a removable, nonvolatile magnetic disk 152, and an optical disk drive 155 that reads from or writes to a removable, nonvolatile optical disk 156 such as a CD ROM or other optical media. Other removable/non-removable, volatile/nonvolatile computer storage media that can be used in the exemplary operating environment include, but are not limited to, magnetic tape cassettes, flash memory cards, digital versatile disks, digital video tape, solid state RAM, solid state ROM, and the like. The hard disk drive 141 is typically connected to the system bus 121 through an non-removable memory interface such as interface 140, and magnetic disk drive 151 and optical disk drive 155 are typically

connected to the system bus 121 by a removable memory interface such as interface 150.

The drives and their associated computer storage media discussed above and illustrated in Fig. 1 provide storage of computer-readable instructions, data structures, program modules, and other data for computer 100. In Fig. 1, for example, hard disk drive 141 is illustrated as storing operating system 144, application programs 145, other program modules 146, and program data 147. Note that these components can either be the same as or different from operating system 134, application programs 135, other program modules 136, and program data 137. Operating system 144, application programs 145, other program modules 146, and program data 147 are given different numbers here to illustrate that, at a minimum, they are different copies. A user may enter commands and information into the computer 100 through input devices such as a keyboard 162 and pointing device 161, commonly referred to as a mouse, trackball, or touch pad. Other input devices (not shown) may include a microphone, joystick, game pad, satellite dish, scanner, or the like. These and other input devices are often connected to the processing unit 120 through a user input interface 160 that is coupled to the system bus, but may be connected by other interface and bus structures, such as a parallel port, game port, or a universal serial bus (USB). A monitor 191 or other type of display device is also connected to the system bus 121 via an interface, such as a video interface 190. In addition to the monitor, computers may also include other peripheral output devices such as speakers 197 and printer 196, which may be connected through an output peripheral interface 195.

The computer may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 180. The remote computer 180 may be a personal computer, a server, a router, a network PC, a peer device or other common network node, and typically includes many or all of the elements described above relative to computer 100, although only a memory storage device 181 has been illustrated in Fig. 1. The logical connections depicted in Fig. 1 include a local area network (LAN) 171 and a wide area network (WAN) 173, but may also include other networks. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets, and the Internet.

When used in a LAN networking environment, the computer 100 is connected to the LAN 171 through a network interface or adapter 170. When used in a WAN networking environment, the computer 100 typically includes a modem 172 or other means for establishing communications over the WAN 173, such as the Internet. The modem 172, which may be internal or external, may be connected to the system bus 121 via the user input interface 160, or other appropriate mechanism. In a networked environment, program modules depicted relative to the computer 100, or portions thereof, may be stored in the remote memory storage device. By way of example, and not limitation, Fig. 1 illustrates remote application programs 185 as residing on memory device 181. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers may be used.

Fig. 2 is a block diagram of a Server-Client system 200 in accordance with the implementations described herein. The system 200 includes a Server computer 202 and a Client computer 204. The Server computer 202 has a processor 206 and

memory 208. The memory 208 stores a page generator 210 for generating web pages, including a web page 212 shown in the memory 208. A page delivery module 214 in the memory 208 delivers the web page 212 to the Client computer 204 via a network (not shown).

The web page 212 contains executable script 216 and a control object 218, which is invoked by the script 216 when the script 216 is executed on the processor 206. A confirmation module 220 is included in the control object 218. As will be discussed in greater detail below, the confirmation module 220 is transmitted to the Client computer 204 with the control object 218 where it authenticates any web page that attempts to invoke the control object 218 and determines if an authenticated source is authorized to invoke the control object 218.

A digital signature module 222 is stored in the memory 208 of the Server computer 202. The digital signature module 222 is configured to digitally sign the web page 212 using any method known in the art. When the web page 212 is digitally signed, a digital signature 226 is attached to the web page 212. The digital signature 226 enables the Client computer 204 to authenticate the source of the web page 212.

Depending on the implementation, the digital signature module 222 may sign each web page generated by the page generator 210, or the digital signature module 222 may only sign web pages that invoke a control. Regardless of the implementation used in the present example, the web page 212 is digitally signed with the digital signature 226 because the web page 212 contains the control object 218 which is invoked by the web page 212.

The control object 218 is a reusable software component that conforms to a standard, such as the COM (common object model) standard. The control object 218 may be used in a variety of containers, such as a Visual Basic program, a C++ program, an HTML web page, etc. The control object 218, when executed, performs a function within the Client computer 204. This function may include, but is not limited to, accessing data, manipulating data, providing animation, displaying objects, etc.

The Client computer 204 includes a processor 227 and memory 228. A web browser 230 is stored in the memory 228 and executes on the processor 227. The web browser 230 enables the Client computer 204 to access the web page 212 on the server 202. As shown in Fig. 2, a copy of the web page 212 (designated as web page 212') has been downloaded to the Client computer 204 and is stored in the memory 228. The downloaded web page 212' includes a script 216' (a copy of the script 216) and a control object 218' (a copy of the control object 218). A copy of the confirmation module 218 (designated as confirmation module 218') has been downloaded with the web page 212' and is a part of the control object 218'. The web page 212' is digitally signed with a digital signature 226' that was downloaded with the web page 212'.

Fig. 3 is a flow diagram of a method to prevent execution of the control object 218' by an unauthorized web page. For this discussion, continuing reference will be made to the elements shown in Fig. 2.

At step 300, the web browser 230 on the Client computer 204 requests a download of the web page 212 from the Server computer 202. If the web page 212 includes script 216 that invokes a control object ("Yes" branch, step 302), then the digital signature module 222 on the Server computer 202 digitally signs

the web page 212 by attaching the digital signature 226 to the web page 212 at step 304. The signed web page 212 is delivered to the Client computer 202 at step 306. If the web page 212 does not invoke a control object ("No" branch, step 302), the web page 212 is delivered to the Client computer 204 at step 306 without a digital signature.

It is noted that step 302 is an optional step. If step 302 is not included in the process, the digital signature module 222 will compute and attach a digital signature to every web page that is downloaded from the Server computer 202. The selected implementation depends on which implementation requires lower requirements of server resources.

At step 308, the Client computer 204 receives the web page 212, 212' from the Server computer 202. On many systems, a user of the Client computer 204 will be notified at this point if the user wishes to download the web page 212 having the control object 218. For purposes of the present discussion, it is assumed that the user downloads the control object 218 with the web page 212.

If a web page or other application attempts to invoke the control object 218' on the Client computer 204 ("Yes" branch, step 310), the confirmation module 220' authenticates the source of the web page 212' at step 312. The confirmation module 220' determines from the digital signature 226' if the web page 212' is from a source the web page 212' purports to come from. The exact method of doing this is well known in the art.

If the confirmation module 220' determines that the web page 212' has come from the source indicated by the web page 212', the confirmation module 220' then determines if the source is an authorized source at step 314. This can be done in several ways. The author of the control object 218' may include a list of

sources that the author trusts to invoke the control object 218', or the user may be prompted at some point by the control object 218' to enter sources which the user trusts to invoke the control object 218' safely, or a list of trusted sites may be stored in the memory of the Client computer 204, etc. With any such implementation, the control object 218' checks the name of the source against one or more source names to determine if the source is authorized to invoke the control object 218'.

It is also noted that, in another implementation, the steps performed by the confirmation module 220' may be performed by the web browser 230 or by a module located in the web browser 230. In such an implementation, when the web page 212' attempts to invoke the control object 218', the web browser 230 will detect or be notified of the event and will attempt to authenticate and authorize the source of the web page 212'.

If the confirmation module 220' determines that the web page 212' has come from an authenticated and authorized source (the Server computer 202 in this example), then the control object 218' is executed at step 318. If the source cannot be authenticated ("No" branch, step 312) or if the source is not authorized to invoke the control object 218' ("No" branch, step 314), then the control object 218' will not be executed.

Conclusion

Control objects embedded in web pages are powerful tools that give a programmer free access to a user's computer. The implementations described provide a user with a way to prevent control objects from being executed by unauthorized users. In this way, the user is assured of the source of the control object and, if the user trusts the source, the user can confidently allow the control object to be invoked.

A user is also assured that once a control object is downloaded to the user's computer, it cannot be invoked by a web page or other application from a source other than the source of the web page or application that originally included the control object.

Although the implementation described herein have been described in language specific to structural features and/or methodological steps, it is to be understood that the invention defined in the appended claims is not necessarily limited to the specific features or steps described. Rather, the specific features and steps are disclosed as preferred implementations.